Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

BAKALÁŘSKÁ PRÁCE

PLZEŇ, 2018 Jakub Matoušek

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: **2017/2018**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Jakub MATOUŠEK**
Osobní číslo:	**A15B0542P**
Studijní program:	**B3918 Aplikované vědy a informatika**
Studijní obor:	**Kybernetika a řídicí technika**
Název tématu:	**Návrh optimálního řízení RC modelu auta pomocí dynamického programování**
Zadávající katedra:	**Katedra kybernetiky**

Zásady pro vypracování:

1. Seznamte se s problematikou modelování pohybu RC modelu auta a se základním principem dynamického programování.
2. Zvolte vhodný nelineární spojitý stavový model popisující pohyb RC modelu auta.
3. Nalezněte diskrétní modely zvoleného spojitého modelu s využitím různých postupů diskretizace. Jeden z nich vyberte a proveďte kvantizaci prostoru vstupů a stavového prostoru.
4. Zvolte matematický popis prostředí, ve kterém se bude RC model auta pohybovat, a navrhněte ztrátovou funkci, která bude reprezentovat cíle řízení.
5. Využijte aproximativní dynamické programování pro návrh regulátoru.
6. Proveďte simulační experimenty a zhodnoťte dosažené výsledky.

Rozsah grafických prací: **dle potřeby**

Rozsah kvalifikační práce: **30-40 stránek A4**

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:

[1] Howie Choset, Kevin M. Lynch, Seth Hutchinson, George a. Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrun. Principles of Robot Motion. MIT Press, Cambridge, MA, USA, 2005.
[2] Rajesh Rajamani. Vehicle Dynamics and Control. Springer, New York, NY, USA, 2 edition, 2012.
[3] Draguna Vrabie, Kyriakos G. Vamvoudakis, and Frank L. Lewis. Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles. The Institution of Engineering and Technology, London, UK, 1 edition, 2013.

Vedoucí bakalářské práce: **Ing. Ivo Punčochář, Ph.D.**
Nové technologie pro informační společnost

Datum zadání bakalářské práce: **1. listopadu 2017**

Termín odevzdání bakalářské práce: **18. května 2018**

Doc. Dr. Ing. Vlasta Radová
děkanka

L.S.

Prof. Ing. Josef Psutka, CSc.
vedoucí katedry

V Plzni dne 1. listopadu 2017

PROHLÁŠENÍ

Předkládám tímto k posouzení a obhajobě bakalářskou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne 16. května 2018

.....
vlastnoruční podpis

Abstrakt

Tato bakalářská práce se zabývá hledáním optimálního řízení pro RC model auta užitím dynamického programování. Nejprve je představen zjednodušený spojitý kinematický model, následně je diskutována problematika diskretizace nelineárního modelu a vhodnost různých diskretizačních metod.

V následující části práce jsou formulovány základy dynamického programování pro řešení dynamických funkcionálních úloh na konečném a nekonečném časovém horizontu, které jsou poté aplikovány na jednoduchý příklad. Poté jsou popsány důvody a možnosti kvantizace stavového prostoru. Nakonec jsou provedeny experimenty hledající optimální řízení v prostředí s překážkami. Součástí práce je také grafické uživatelské rozhraní, navržené v Matlabu, pro vizualizaci pohybu auta v diskrétním a spojitém prostoru.

Klíčová slova: dynamické programování, optimální řízení, RC model auta

Abstract

This bachelor thesis deals with optimal control of RC car model using dynamic programming. At first, the simplified continuous kinematic model is introduced. Further, the issue of non-linear model discretization methods and their utility is discussed.

Following part of the thesis focuses on formulation of dynamic programming for solving dynamical functional problems on finite and infinite time horizon, which are then applied in a simple example. After that, reasons and techniques of state space quantization are presented. Finally experiments for finding an optimal control in surrounding environment are presented. A graphical user interface designed in Matlab for visualization of a car movement in discrete and continuous space is briefly described.

Keywords: dynamic programming, optimal control, RC car model

PODĚKOVÁNÍ

Rád bych poděkoval Ing. Ivo Punčochářovi, Ph.D za pomoc a podporu při vedení bakalářské práce. Mé poděkováni patři též Ing. Janu Škachovi za vedení projektů předcházejících bakalářské práci. Nakonec děkuji Ing. Miroslavu Flídrovi, Ph.D za přípravu RC modelu auta pro měření v systému VICON.

Obsah

Kapitola 1

Úvod

Autonomní řízení osobních i nákladních automobilů v současnosti přitahuje čím dál větší pozornost. Velké společnosti jako Google nebo Tesla investují nemalé prostředky na vývoj těchto autonomních řídících systémů. Za zjednodušenou úlohu zabývající se tímto problémem může být považováno řízení modelu RC auta bez lidského zásahu.

Pro návrh takového řízení je nejprve nutné ověřit model pohybu auta. Poté je třeba použít systém pro určování polohy RC auta a tvaru trati, například VICON, kde je již implementováno sledování objektů a není třeba řešit rozpoznávání obrazu. Dalším krokem je nalezení optimální trati, po které by model RC auta měl jet. Nakonec bude třeba navrhnout algoritmus řízení a ten propojit s reálným modelem RC auta, použitím Raspberry Pi nebo Arduina.

Cílem této práce je ověřit funkčnost dynamického programování pro nekonečný časový horizont při hledání optimálního řízení RC autíčka. Zadání pro řízení je dostat se z bodu A do bodu B a vyhnout se při tom překážkám, jejichž umístění je známo.

Kapitola 2

Model RC auta

V této kapitole je představen reálný model RC auta a jeho základní parametry. Následně je popsán jednoduchým spojitým nelineárním kinematickým modelem. Poté je diskutována problematika diskretizace tohoto modelu. A na závěr je porovnána použitelnost několika základních diskretizačních metod pro problém hledání optimální strategie řízení.

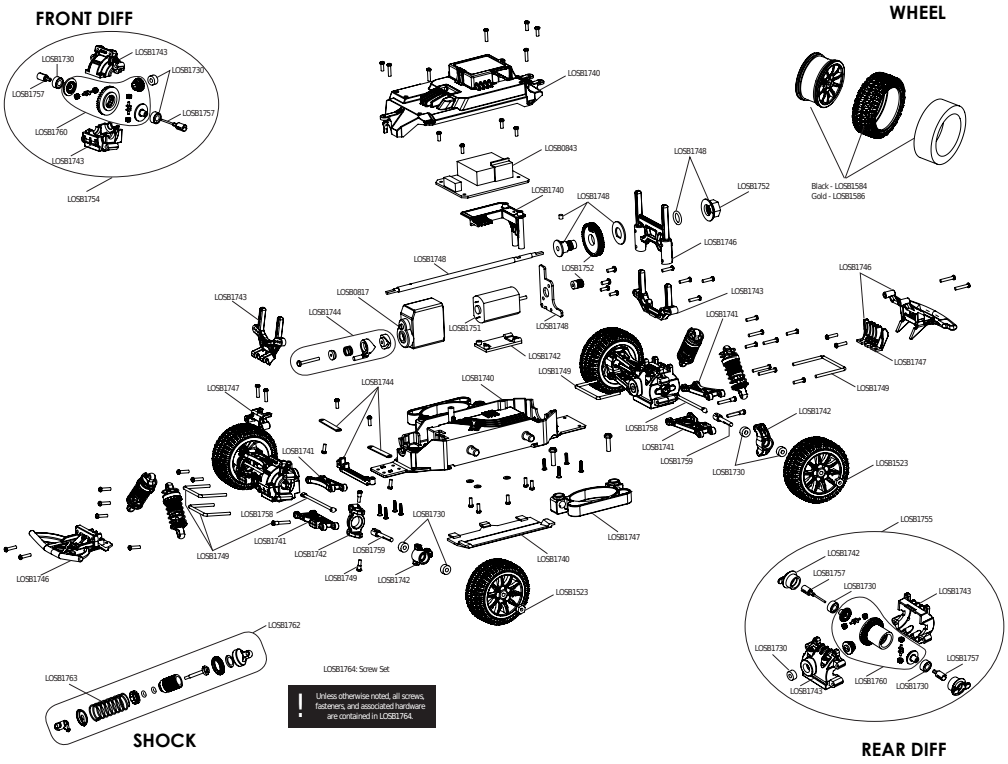
2.1 Fyzický model

V této práci je cílovou aplikací automatické řízení reálného modelu RC auta. Konkrétně se jedná o model Micro-Rally Car od firmy Losi(LOSB0241IT1) s pohonem na všechna čtyři kola. Tento model je v měřítku 1:24. Fotografie RC modelu auta společně s dálkovým ovladačem je uvedena na obrázku 2.1, součástky modelu jsou poté na obrázku 2.2 Podstatné technické parametry RC modelu byly určeny z katalogového listu a přímým měřením na reálném modelu. Manuálně řízený pohyb RC modelu byl rovněž zaznamenán pomocí systému pro snímání pohybu VICON. Maximální úhel natočení kol řízené přední nápravy byl změřen pomocí úhloměru. Zjištěné údaje jsou shrnuty v tabulce 2.1.

Tabulka 2.1: Technické specifikace

Hmotnost[g]	160
Rozchod[cm]	9
Rozvor[cm]	11
Max. úhel natočení kol[rad]	0.3
Max. rychlost[m/s]	4.5
Vnější rozměřy[cm]	163x95x99

Obrázek 2.1: Obrázek RC modelu

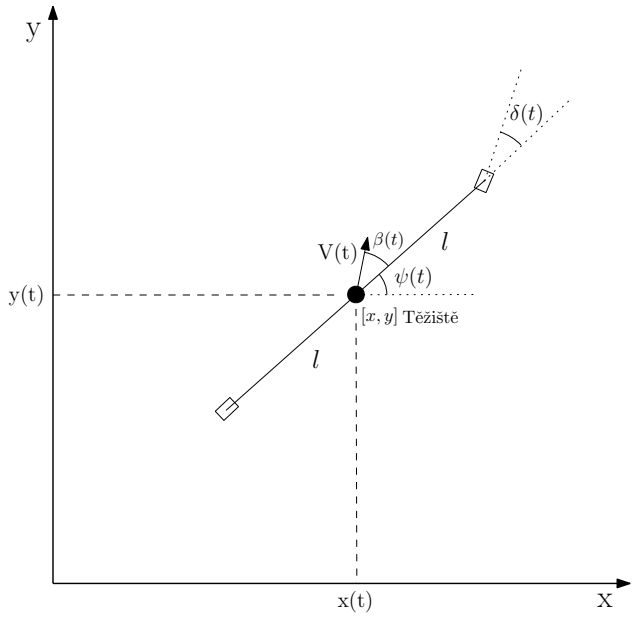


Obrázek 2.2: Součástky RC modelu

2.2 Matematický model

Pro specifikaci problému hledání optimálního řízení a jeho řešení použitím technik dynamického programování (viz kapitola 3) je třeba nalézt matematický model pohybu RC auta, který bude dále použit pro návrh a simulační otestování regulátoru. Hledáme tedy soustavu diferenciálních rovnic popisující pohyb RC auta. Hledání těchto rovnic může být pojato dvěma způsoby. První možností je použít úplný dynamický model zahrnující všechny působící síly, momenty sil a vlastnosti materiálů, tímto komplexním modelováním se zabývá např. [5]. Druhou možností je použít zjednodušený kinematický model, tímto modelováním se zabývá např. [4] v kapitole Lateral vehicle dynamics, poněkud složitější model lze potom nalézt v [2].

Dále bude uvažován zjednodušený kinematický model pohybu RC auta (viz obrázek 2.3), který odpovídá modelu jízdního kola [4] a je pro potřeby této práce postačující. Zjednodušení spočívá v nahrazení dvou předních (resp. dvou zadních) kol jedním kolem uprostřed. Tento model popisuje jak se v čase vyvíjí poloha těžiště RC auta v souřadné soustavě, úhel natočení podélné osy tohoto auta od x-ové osy souřadného systému a úhel skluzu, tj. úhel, který svírá vektor rychlosti těžiště s podélnou osou modelu RC auta v závislosti na zadaném natočení předního kola a velikosti rychlosti těžiště.



Obrázek 2.3: Geometrická interpretace kinematického modelu

Spojitý kinematický model je dán soustavou nelineárních diferenciálních

rovnic

$$\begin{aligned}\dot{x}(t) &= V(t)cos(\psi(t)+\beta(t)),\\ \dot{y}(t) &= V(t)sin(\psi(t)+\beta(t)),\\ \dot{\psi}(t) &= \frac{V(t)cos(\beta(t))}{2l}tan(\delta(t)),\end{aligned} \tag{2.1}$$

kde $x(t)$[m] a $y(t)$[m] reprezentují souřadnice těžiště RC modelu, $\psi(t)$[rad] je směrový úhel tohoto auta, l[m] je vzdálenost mezi předním(resp. zadním) kolem a těžištěm, $\delta(t)$[rad] je úhel natočení předního kola vůči podélné ose, $V(t)$[m/s] je velikost rychlosti RC auta a $\beta(\delta(t))$[rad] je úhel skluzu daný vztahem

$$\beta(\delta(t)) = tan^{-1}\left(\frac{tan(\delta(t))}{2}\right).$$

Jedná se tedy o stavový model, kde $[x(t), y(t), \psi(t)]$ je vektor stavu a vektor $[V(t), \delta(t)]$ se skládá ze vstupů modelu. Předpokládá se, že stav je v každém okamžiku známý.

2.3 Diskretizace

Výše zmíněný stavový model (2.1) je třeba diskretizovat, aby bylo možné, při hledání optimálního řízení, navrhnout diskrétní regulátor. Mějme spojitý stavový model daný soustavou diferenciálních rovnic

$$\dot{\mathbf{x}}(t) = \mathbf{f}_c(\mathbf{x}(t), \mathbf{u}(t)),$$

kde $\mathbf{x}(t) \in \mathbb{R}^{n_x}$ je stav, $\mathbf{u}(t) \in \mathbb{R}^{n_u}$ je vstup, $\mathbf{f}_c : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \mapsto \mathbb{R}^{n_x}$ je daná funkce a $\mathbf{x}(t_0)$ je daná počáteční podmínka. Cílem diskretizace je nalézt funkci $\mathbf{f}_d$ pro diskrétní stavový model ve tvaru

$$\mathbf{x}_{k+1} = \mathbf{f}_d(\mathbf{x}_k, \mathbf{u}_k)$$

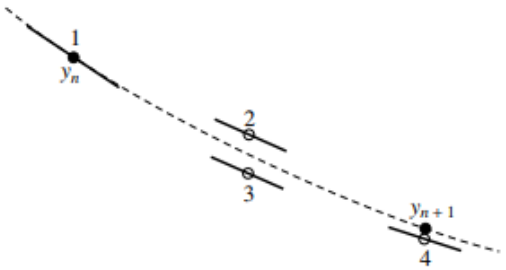
tak, aby $\mathbf{x}_k = \mathbf{x}(t_k)$, pro všechna $t_k = t_0 + kT_S$, T_s je perioda vzorkování a $k \in \{0, 1, 2, ...\}$. Při hledání $\mathbf{f}_d$ se předpokládá, že vstup je v rámci vzorkovací periody konstantní. Primárně se tedy jedná o nalezení řešení soustavy nelineárních diferenciálních rovnic. Exaktní řešení lze standardními matematickými metodami ovšem nalézt jen pro lineární a některé speciální nelineární modely. Pro účely návrhu regulátoru bude postačovat aproximace $\mathbf{x}_k \approx \mathbf{x}(t_k)$. Výpočet $\mathbf{x}_k$ může být reprezentován buď numerickou metodou, nebo funkčním předpisem. Dále následuje porovnání několika základních diskretizačních technik pro nalezení funkčních předpisů.

2.3.1 Runge-Kuttovy metody

Runge-Kuttovy metody jsou explicitní a implicitní iterativní metody pro hledání aproximativních řešení obyčejných diferenciálních rovnic. Vyšší řády těchto metod vyhodnocují derivaci ve vícero postupných bodech (čtvrtého řádu ve čtyřech bodech atd.) [3]. Na obrázku 2.4 lze vidět evaluaci pro Runge-Kuttovu metodu čtvrtého řádu. Obecně lze Runge-Kuttovy metody zapsat takto

$$y_{n+1} = y_n + h\sum_{i=1}^{p}\omega_i k_i,$$
$$k_i = f(t + \alpha_i h, y_n + h\sum_{j=1}^{i-1}\beta_{ij}k_j),$$

kde α, β, ω, jsou předem dané koeficienty, h je časový krok a $i = 1, 2..I$, I se zvyšuje s rostoucím řádem metody.



Obrázek 2.4: Evaluace sklonu [3]

Eulerova metoda (Runge-Kuttova I. řádu)

Tato metoda využívá definice derivace. Pro zvolenou periodu vzorkování T_s tedy aproximace derivace vypadá takto

$$\dot{x}(t) = \lim_{T_s \to 0}\frac{x(t+T_s) - x(t)}{T_s},$$
$$\dot{x}(t) \approx \frac{x_{k+1} - x_k}{T_s}. \tag{2.2}$$

Použitím Eulerovy dopředné metody lze pohyb RC auta v každém časovém okamžiku $k \in \{0, 1, 2, ...\}$ popsat diskrétním stavovým modelem

$$
\begin{aligned}
x_{1,k+1} &= u_{2,k} \cdot \cos(x_{3,k} + \beta_k) \cdot T_s + x_{1,k}, \\
x_{2,k+1} &= u_{2,k} \cdot \sin(x_{3,k} + \beta_k) \cdot T_s + x_{2,k}, \\
x_{3,k+1} &= \frac{u_{2,k} \cdot \cos(\beta_k)}{2l} \tan(u_{1,k}) \cdot T_s + x_{3,k}, \\
\beta_k &= \tan^{-1}\left(\frac{\tan(u_{1,k})}{2}\right),
\end{aligned} \tag{2.3}
$$

kde $x_{1,k}$[m] je poloha těžiště autíčka v ose x, $x_{2,k}$[m] je poloha těžiště autíčka v ose y, $x_{3,k}$[rad] je úhel natočení autíčka vzhledem k inerciální vztažné soustavě, β_k[rad] je úhel skluzu, l[m] je vzdálenost předního a zadního kola, $u_{1,k}$[rad] je úhel natočení předních kol, $u_{2,k}$[m/s] je rychlost a T_s[s] je perioda vzorkování. Stav systému je $\mathbf{x}_k = [x_{1,k}, x_{2,k}, x_{3,k}]^T$ a vstup systému je $\mathbf{u}_k = [u_{1,k}, u_{2,k}]^T$.

Runge-Kuttova II. řádu

Mějme obyčejnou diferenciální rovnici prvního řádu

$$
\dot{x} = f\Big(x(t), u(t)\Big). \tag{2.4}
$$

Potom podle Runge-Kutta jsou vzorce pro výpočet[1]

$$
\begin{aligned}
\alpha_1 &= f\Big(x(t_0), t_0\Big), \\
y(t_0 + h) &= x(t_0) + \alpha_1 h, \\
\alpha_2 &= f\Big(y(t_0 + h), t_0 + h\Big), \\
x(t_0 + h) &= x(t_0) + h\left(\frac{\alpha_1 + \alpha_2}{2}\right).
\end{aligned}
$$

Aplikací těchto vztahů na soustavu diferenciálních rovnic 2.1 získáme vztahy

$$
\begin{aligned}
\beta_k &= \tan^{-1}\left(\frac{\tan(u_{1,k})}{2}\right), \\
\alpha_{31} &= \frac{u_{2,k} \cdot \cos(\beta_k)}{2l} \tan(u_{1,k}), \\
\alpha_{11} &= u_{2,k} \cdot \cos(x_{3,k} + \beta_k), \\
\alpha_{21} &= u_{2,k} \cdot \sin(x_{3,k} + \beta_k),
\end{aligned}
$$

$$
\begin{aligned}
y_3 &= x_{3,k} + \alpha_{31} \cdot T_s, & \alpha_{32} &= \frac{u_{2,k} \cdot \cos(\beta_k)}{2l} \tan(u_{1,k}), \\
y_1 &= x_{1,k} + \alpha_{11} \cdot T_s, & \alpha_{12} &= u_{2,k} \cdot \cos(y_{3,k} + \beta_k), \\
y_2 &= x_{2,k} + \alpha_{21} \cdot T_s, & \alpha_{22} &= u_{2,k} \cdot \sin(y_{3,k} + \beta_k),
\end{aligned}
$$

$$
\begin{aligned}
x_{3,k+1} &= x_{3,k} + T_s \left(\frac{\alpha_{31} + \alpha_{32}}{2} \right), \\
x_{1,k+1} &= x_{1,k} + T_s \left(\frac{\alpha_{11} + \alpha_{12}}{2} \right), \\
x_{2,k+1} &= x_{2,k} + T_s \left(\frac{\alpha_{21} + \alpha_{22}}{2} \right).
\end{aligned}
$$

ODE45

Matlab ODE45 solver používá explicitní metodu RK(4,5), nazývanou rovněž Dormand-Prince pair, dle jejích autorů. Jedná se o jednokrokovou metodu, která je založena na vyhodnocení pravé strany diferenciální rovnice v sedmi bodech. Rozdíl v řešení při použití čtvrtého a pátého řádu je pak použit k nastavení adaptivního kroku diskretizace [3], [6].

2.3.2 Porovnání

Přesnost metod diskretizace spojitého modelu byla porovnána proti Matlab ODE45 solveru s velmi přesným řešením, které bylo získáno nastavením absolutní tolerance na $1 \cdot 10^{-7}$ a maximálního kroku na $1 \cdot 10^{-3}$.

Metody byly testovány pro vzdálenost kol $l = 0.055$[m], konstantní rychlost $V = 4$[m/s], úhel natočení koleček se v průběhu simulace mění lineárně od 0[rad] do 0.3[rad] a poté do -0.3[rad], viz obrázek 2.5. Simulace trvala čtyři sekundy se 400 vzorkovacími okamžiky (tzn. krok simulace pro Runge-Kuttovy metody byl 0.1). Grafické porovnání přesnosti řešení je uvedeno na obrázku 2.6. Na obrázcích 2.7 a 2.8 jsou uvedeny detaily z konce trajektorie, jak je naznačeno na obrázku 2.6 pomocí černých obdélníků. V tabulce 2.2 jsou pak uvedeny výsledky srovnání číselně. Pro srovnání byla vybrána maximální hodnota absolutní chyby v průběhu celé simulace pro jednotlivé stavy

$$
l_i^{max} = \max_{k \in \{1,2 \ldots K\}} \mid \mathbf{x}_{k,i}^p - \mathbf{x}_{k,i} \mid,
$$

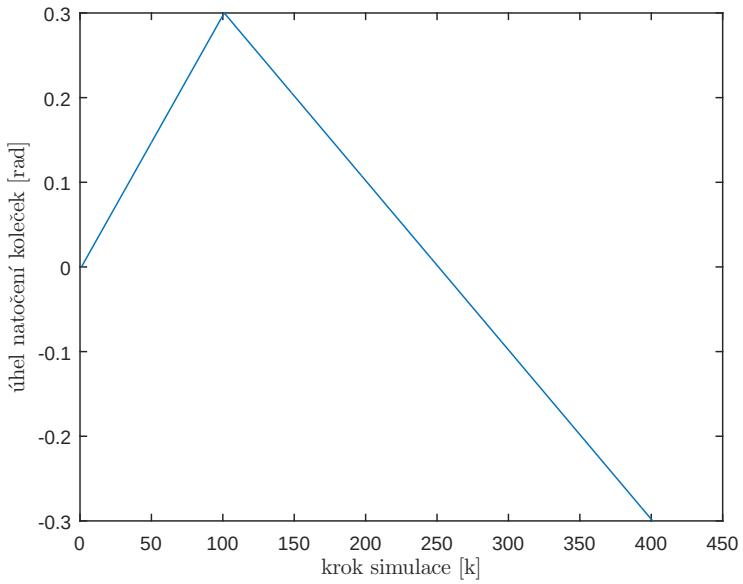
průměrná chyba na jeden vzorkovací okamžik

$$
l_i^p = \frac{\sum_{k=0}^{K} \mid \mathbf{x}_{k,i}^p - \mathbf{x}_{k,i} \mid}{K}
$$

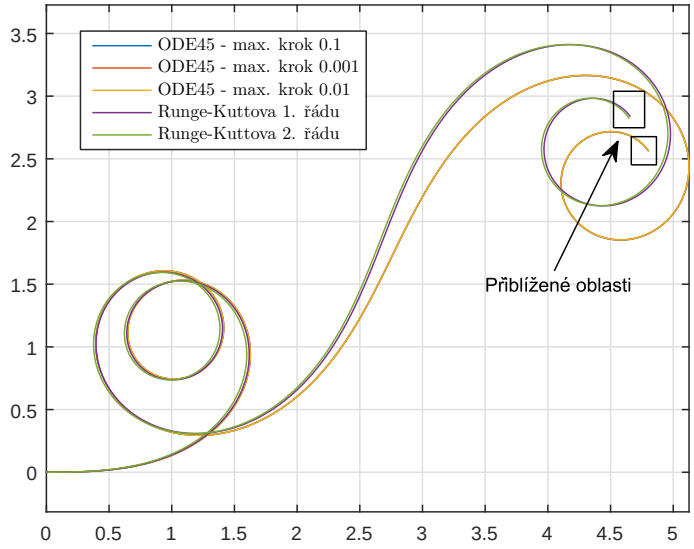
a časové nároky na výpočet celé dráhy T_t. V předchozích vztazích $k = \{0, 1...400\}$, $\mathbf{x}^p$ označuje přesné řešení z ODE45 (viz. výše) a $\mathbf{x}$ označuje porovnávané řešení, počítáno pro každou složku zvlášť, čili $\mathbf{l}^{max} \in \mathbb{R}^{n_x}$ a $\mathbf{l}^p \in \mathbb{R}^{n_x}$. Hodnoty uvedené v tabulce 2.2 jsou zaokrouhleny na čtyři desetinná místa a jsou v pořadí pro stavy x_1, x_2, x_3.

Tabulka 2.2: Porovnání diskretizačních metod

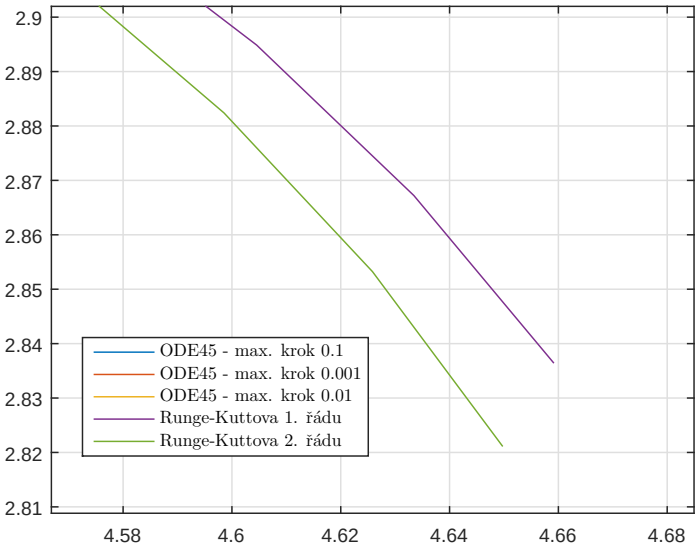
	$\mathbf{l}^{max}$	$\mathbf{l}^p$	$T_t[s]$
Runge-Kuttova 1. řádu	0.1912 0.2975 0.0902	0.0662 0.1064 0.0573	0.2
Runge-Kuttova 2. řádu	0.1919 0.2790 0.0902	0.0791 0.1153 0.0573	0.4
ODE45 (max. krok $1 \cdot 10^{-1}$)	0.0007 0.0013 0.0004	0.0002 0.0005 0.0002	0.3
ODE45 (max. krok $1 \cdot 10^{-2}$)	0.0 0.0 0.0	0.0 0.0 0.0	1.0



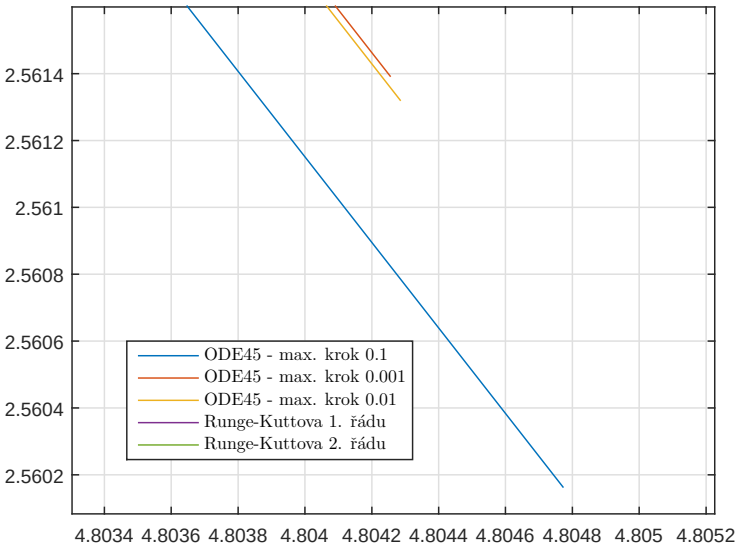
Obrázek 2.5: Úhel natočení koleček při testování diskretizací



Obrázek 2.6: Dráha pro různé diskretizační metody



Obrázek 2.7: Detail konce dráhy pro různé diskretizační metody



Obrázek 2.8: Detail konce dráhy pro různé diskretizační metody

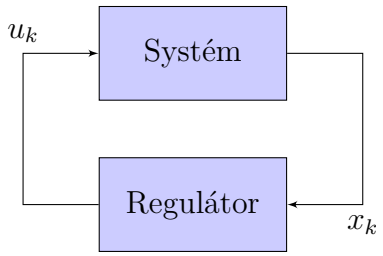
2.3.3 Vybraná metoda

Vzhledem k náročnosti výpočtu optimálního řízení pomocí dynamického programování je časová náročnost simulace modelu jedním z hlavních měřítek při výběru diskretizační metody. Jelikož se tato práce zabývá pouze teoretickým ověřením funkčnosti, nevyžaduje velkou shodu spojitého a diskretizovaného modelu. Proto se v tomto případě jeví jako nejlepší Runge-Kuttovy metody. ODE45 solver nastavený na maximální velikost kroku diskretizace 0.1 není sice o moc více časově náročnější, ale v algoritmu iterace váhové funkce by musel být volán ve for cyklu, zatímco funkce modelu autíčka může být volána jen jednou pro všechny stavy najednou. Výše zmíněný ODE45 solver(případně přesnější) by bylo pravděpodobně třeba zvolit pro výpočet řízení reálného RC auta, kdy by již přesnost diskretizovaného modelu hrála větší roli.

Kapitola 3

Dynamické programování

Tato práce je zaměřena na použití dynamického programování pro řešení dynamických funkcionálních úloh na konečném nebo nekonečném časovém horizontu [7].[1] Hlavní myšlenkou této techniky je rozdělení složitého problému na jednodušší podproblémy s využitím rekurze. Toto rekurzivní rozložení problému je reprezentováno Bellmanovou funkcionální rovnicí, kterou lze řešit různými způsoby, podle toho, zda je problém definován na konečném nebo nekonečném časovém horizontu. Řešením Bellmanovy rovnice je Bellmanova funkce, která popisuje hodnoty budoucích ztrát. S jejím využitím lze pro libovolný stav určit optimální vstup pomocí jednoduché statické optimalizace. Na obrázku 3 lze vidět jednoduchou regulační smyčku, kde systém je řízen regulátorem, který počítá optimální řízení na základě aktuálního stavu a řešení Bellmanovy rovnice.



Obrázek 3.1: Diagram řízení

[1] Technika dynamického programování se používá i při řešení úloh, které ve své podstatě nejsou dynamické, například sázení textu programem LaTeX(Lamport).

3.1 Optimální řízení na konečném časovém horizontu

Mějme systém popsaný stavovým modelem

$$\mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k, \mathbf{u}_k),$$

kde $\mathbf{x}_k \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$ je stav systému, $\mathbf{u}_k \in \mathcal{U} \subseteq \mathbb{R}^{n_u}$ je řízení a k je diskrétní časový okamžik, $\mathbf{f}_k$ je známa funkce a $\mathbf{x}_0$ známý počáteční stav. Cílem je navrhnout regulátor na konečném časovém horizontu, $k = 0, 1...F$

$$\mathbf{u}_k = \boldsymbol{\gamma}_k(\mathbf{x}_k), \tag{3.1}$$

kde $\boldsymbol{\gamma}_k : \mathcal{X} \mapsto \mathcal{U}$ je neznámá funkce, která reprezentuje strategii řízení v čase k. Definujme ztrátovou funkci $L_k^c(\mathbf{x}_k, \mathbf{u}_k), L_k^c : \mathcal{X} \times \mathcal{U} \mapsto \mathbb{R}^+$, která určuje ztrátu (cenu), která vznikne při použití řízení $\mathbf{u}_k$ ve stavu $\mathbf{x}_k$. Funkce L_k^c může být například kvadratická funkce stavu a řízení. Mějme následující kritérium kvality řízení

$$J(\mathbf{x}_0, \boldsymbol{\gamma}_0^F) = \sum_{k=0}^{F} L_k^c(\mathbf{x_k}, \mathbf{u_k}), \tag{3.2}$$

které hodnotí regulátor na konečném časovém intervalu.

Optimální posloupnost řízení regulátorů má takovou vlastnost, že pro každý stav $\mathbf{x}_k$ a každou předchozí posloupnost řízení $\mathbf{u}_0^{k-1}$ je následující posloupnost řízení optimální pro stav $\mathbf{x}_k$ a zbývající(budoucí) ztráty jsou minimální.

Předpokládejme, že známe optimální regulátor pro koncovou část horizontu řízení $\boldsymbol{\gamma}_{k+1}^{\star F}$. Bellmanova rekurzivní rovnice lze neformálně odvodit následujícím způsobem. Rozdělme celý horizont řízení na dvě části: časové okamžiky $0, 1, ..., k$ a časové okamžiky $k+1, ..., F$, kde k je aktuální časový okamžik. Zaveďme

$$V_{k+1}^{\star}(\mathbf{x}_{k+1}) = \sum_{l=k+1}^{F} L_k^c\left(\mathbf{x}_l, \boldsymbol{\gamma}_l^{\star}(\mathbf{x}_l)\right), \tag{3.3}$$

kde $V_{k+1}^{\star} : \mathcal{X} \mapsto \mathbb{R}^+$ je Bellmanova funkce. Nyní se budeme snažit najít Bellmanovu funkci pro časový okamžik k. Jelikož $V_{k+1}^{\star}$ odpovídá minimálním možným budoucím ztrátám lze Bellmanovu funkci pro časový okamžik k zapsat jako

$$V_k^{\star}(\mathbf{x}_k) = \min_{\overline{\boldsymbol{u}} \in U} \left\{ L_k^c(\mathbf{x}_k, \overline{\boldsymbol{u}}) + V_{k+1}^{\star}\Big(\mathbf{f}_k(\mathbf{x}_k, \overline{\boldsymbol{u}})\Big) \right\}. \tag{3.4}$$

Počáteční podmínka pro rekurzi je $V^{\star}_{F+1} = 0$. Optimální regulátor pro časový okamžik k je tedy

$$\mathbf{u}_k = \boldsymbol{\gamma}^{\star}_k(\mathbf{x}_k) = \underset{\overline{u}\in U}{\operatorname{argmin}} \left\{ L^c_k(\mathbf{x}_k, \overline{\boldsymbol{u}}) + V^{\star}_{k+1}\Big(\mathbf{f}_k(\mathbf{x}_k, \overline{\boldsymbol{u}})\Big) \right\}. \tag{3.5}$$

Při praktickém řešení (3.4) a (3.5) se potýkáme s problémem efektivní reprezentace Bellmanovy funkce V_k a exaktní řešení lze jednoduše nalézt pouze ve speciálních případech. Jedním z nich je LQ úloha, kdy V_k je kvadratická forma, dalším případem je situace, kdy jsou množiny $\mathcal{X}$ a $\mathcal{U}$ diskrétní a funkci V_k lze reprezentovat tabulkou. Příkladem takového problému může být například hledání nejrychlejší/nejkratší cesty skrz jednoduchou mapu.

3.2 Optimální řízení na nekonečném časovém horizontu

V některých případech není formulace úlohy na konečném časovém horizontu vhodná a to pokud je samotný problém definován na nekonečném časovém horizontu nebo že horizont je sice konečný, ale velmi dlouhý, což by vedlo ke značným paměťovým nárokům regulátoru. Obecné řešení neomezené časem může být navrženo použitím Bellmanovy funkcionální rovnice.

Aby úloha optimálního řízení na nekonečném časovém horizontu dávala smysl, musí být systém a kritérium t-invariantní, tj. $\mathbf{f}$ a L^c místo $\mathbf{f}_k$ a L^c_k. Z hlediska zjednodušení teoretické analýzy je vhodné, aby bylo kritérium na nekonečném horizontu konečné pro dostatečně širokou třídu regulátorů, proto se v případě nekonečného horizontu často používá diskontní kritérium

$$J(\mathbf{x}_0, \boldsymbol{\gamma}_0^{\infty}) = \lim_{F\to\infty} \sum_{k=0}^{F} \eta^k L^c(\mathbf{x}_k, \mathbf{u}_k), \tag{3.6}$$

kde $\eta \in (0, 1)$ je diskontní faktor, který redukuje důležitost budoucích ztrát a zaručuje existenci limity v (3.6) za předpokladu, že L^c je shora omezená funkce.

Potom je optimální strategie řízení dána řešením Bellmanovi rovnice optimality

$$V^{\star}(\mathbf{x}_k) = \min_{\overline{\boldsymbol{u}}\in\mathcal{U}} \left\{ L^c(\mathbf{x}_k, \overline{\boldsymbol{u}}) + \eta V^{\star}\Big(\mathbf{f}(\mathbf{x}_k, \overline{\boldsymbol{u}})\Big) \right\}, \tag{3.7}$$

kde $V^{\star} : \mathcal{X} \to \mathbb{R}$ je Bellmanova funkce. Optimální regulátor pro časový okamžik k je tedy

$$\mathbf{u}_k = \boldsymbol{\gamma}^{\star}(\mathbf{x}_k) = \underset{\overline{u}\in U}{\operatorname{argmin}} \left\{ L^c(\mathbf{x}_k, \overline{\boldsymbol{u}}) + \eta V^{\star}\Big(\mathbf{f}(\mathbf{x}_k, \overline{\boldsymbol{u}})\Big) \right\}. \tag{3.8}$$

Na rozdíl od problému s konečným časovým horizontem je u tohoto problému hledána pouze jedna funkce $V^\star$ a optimální regulátor je t-invariantní.

3.2.1 Iterace váhové funkce

Vztah (3.7) je nelineární funkcionální rovnice, jejíž řešení lze exaktně nalézt pouze v některých speciálních případech. Pro diskrétní $\mathcal{X}$ a $\mathcal{U}$ se dá nalézt numerické řešení pomocí několika metod: iterace váhové funkce, iterace strategie a zobecněná iterace strategie[7]. V této práci je využita iterace váhové funkce, která je dána vztahem

$$\overline{V}^{(i+1)}(\overline{\mathbf{x}}) = \min_{\overline{\boldsymbol{u}} \in \mathcal{U}} \left\{ L^c(\overline{\mathbf{x}}, \overline{\boldsymbol{u}}) + \eta \overline{V}^{(i)}\Big(\mathbf{f}(\overline{\mathbf{x}}, \overline{\boldsymbol{u}})\Big) \right\}, \tag{3.9}$$

kde $i = 0, 1, ...$ je iterační index. Počáteční funkce $\overline{V}^{(0)}$ může být zvolena jako identicky nulová, nebo, pokud máme k dispozici nějaký odhad Bellmanovy funkce, může být tento použit jako počáteční podmínku pro iteraci váhove funkce. Algoritmus iterace váhove funkce je ukončen po splnění ukončovací podmínky

$$\left\| \overline{V}^{(i)} - \overline{V}^{(i+1)} \right\|_\infty \leq \delta, \tag{3.10}$$

kde $\delta \in \mathbb{R}^+$ je uživatelem zvolená hodnota, ovlivňující přesnost řešení, $\|\cdot\|_\infty$ reprezentuje nekonečnou normu na prostoru funkcí $V : \mathcal{X} \mapsto \mathbb{R}^+$, která je definována jako

$$\|V\|_\infty = \sup_{\mathbf{x} \in \mathcal{X}} |V(\mathbf{x})|.$$

Posloupnost funkcí $\overline{V}^{(i)}$ za jistých předpokladů konverguje k funkci $V^\star$ [7].

Ilustrace algoritmu iterace váhové funkce na příkladu

Návrh optimálního regulátoru pro nekonečný horizont iterací váhové funkce bude ilustrováno na jednoduchém příkladu s konečným počtem stavů a řízení.

Uvažujme devět stavů $x_k \in \mathcal{X} = \{1, 2, ..., 9\}$ které jsou uspořádány do políček 3x3 šachovnice, jak je uvedeno na obrázku 3.2.

1	2	3
4	5	6
7	8	9

Obrázek 3.2: Stavový prostor

Hodnoty řízení $u_k \in \mathcal{U} = \{1, 2...5\}$ reprezentují posuny mezi stavy na šachovnici ve vertikálním a horizontálním směru o jedno políčko:

$$1 : \text{doleva},\ 2 : \text{nahoru},\ 3 : \text{doprava},\ 4 : \text{dolů},\ 5 : \text{bez akce}.$$

Ztrátová funkce $L^c(\mathbf{x}_k, \mathbf{u}_k)$ je definována tabulkou 3.1.

Tabulka 3.1: Tabulka ztrátové funkce

x_k \ u_k	1	2	3	4	5
1	∞	∞	2	6	4
2	8	∞	7	1	2
3	3	∞	∞	5	5
4	∞	7	2	9	1
5	8	9	7	8	0
6	3	9	∞	8	4
7	∞	3	4	∞	6
8	7	1	9	∞	3
9	4	2	∞	∞	6

Čísla v tabulce reprezentují ztrátu při použití řízení u_k ve stavu x_k.
Přechodová funkce systému $f(x_k, u_k)$ je definována tabulkou 3.2.

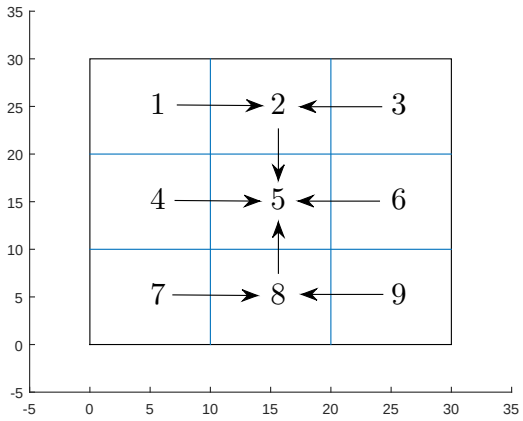
Tabulka 3.2: Tabulka přechodů mezi stavy

x_k \ u_k	1	2	3	4	5
1	-	-	2	4	1
2	1	-	3	5	2
3	1	-	-	6	3
4	-	1	5	7	4
5	4	2	6	8	5
6	5	3	-	9	6
7	-	4	8	-	7
8	7	5	9	-	3
9	8	6	-	-	9

Čísla v tabulce reprezentují index nového stavu x_{k+1}, do kterého systém přejde ze stavu x_k aplikací řízení u_k a "-"značí neplatné řízení v daném stavu
.

Abychom vyřešili tento příklad, musíme nalézt optimální řízení minimalizující součet ztrát pro každé použité řízení na nekonečném časovém horizontu $k = 1, 2....$ Pro tento problém neexistuje analytické řešení Bellmanovy rovnice, proto bylo přibližné řešení nalezeno pomocí iterace váhové funkce.

Prahová hodnota pro zastavovací podmínku byla zvolena $\delta = 1$, diskontní faktor může být, díky jednoduchosti příkladu a existenci řízení "bez akce", zvolen $\eta = 1$, váhová funkce $\overline{V}^{(0)}$ byla inicializována na samé nuly. Algoritmus provedl čtyři iterace. Na obrázku 3.3 je vizualizace optimální strategie řízení γ. Šipky představují optimální řízení pro jednotlivé stavy.

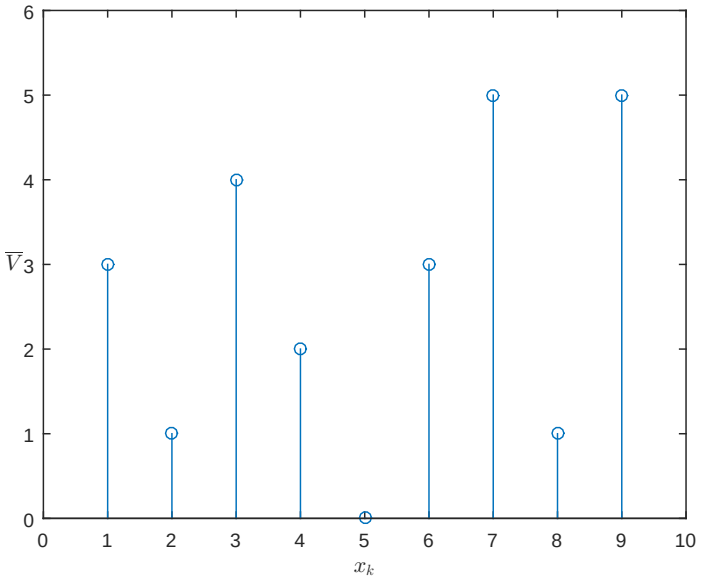


Obrázek 3.3: Strategie řízení

Konečná podoba Bellmanovy funkce je

$$\overline{V}^{(4)} = \Big[3, 1, 4, 2, 0, 3, 5, 1, 5\Big],$$

kde čísla ve vektoru představují hodnoty Bellmanovy funkce pro jednotlivé stavy, viz obrázek 3.4. Optimální regulátor poté generuje řízení na základě aktuálního stavu dle vztahu (3.5).



Obrázek 3.4: Konečná podoba Bellmanovy funkce

Kapitola 4

Návrh regulátoru

4.1 Kvantizace

Jak již bylo zmíněno v kapitole 3.1, je v případě spojitého stavového prostoru zásadním problémem reprezentace Bellmanovy funkce. Jednou z jednoduchých možností řešení je provést kvantizaci stavového prostoru a prostoru vstupů a poté nalézt Bellmanovu funkci pro tento kvantizovaný problém. Ta pak může sloužit jako aproximativní řešení původního problému.

Kvantizace je technika, mapující spojité množiny na diskrétní. Spojitý stavový prostor může být kvantifikován na diskrétní použitím agregační funkce. Obecně tato funkce může vypadat například takto

$$\overline{\mathbf{x}}_k = g(\mathbf{x}_k) = \arg\min_{\overline{\mathbf{x}} \in \overline{\mathcal{X}}} \|\mathbf{x}_k - \overline{\mathbf{x}}\|_2,$$
$$\arg\min_{\overline{\mathbf{x}} \in \overline{X}} = \sqrt{\sum_{i=1}^{n} (x_{k,i} - \overline{x}_i)^2}, \tag{4.1}$$

kde $g : \mathcal{X} \to \overline{\mathcal{X}}$, $\overline{\mathcal{X}}$ je prostor diskrétních stavů, $\overline{\mathbf{x}}$ je diskrétní stav, $\mathbf{x}$ spojitý stav a n je počet stavových proměnných.

Za předpokladu, že diskrétní stavy tvoří rovnoměrnou mřížku ve stavovém prostoru viz obrázek 4.3, lze agregační funkci realizovat, s ohledem na reprezentaci stavového prostoru v Matlabu, po složkách následujícím způsobem

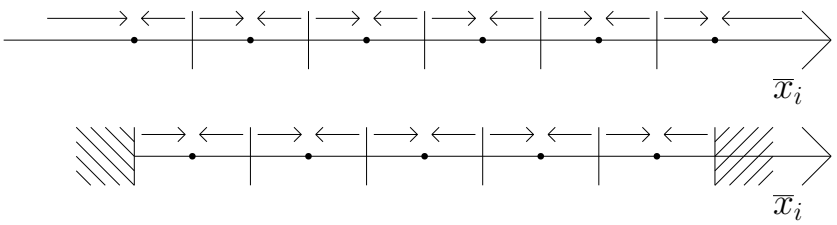
$$\overline{\mathbf{x}_{ind}} = \left\lfloor \frac{\mathbf{x} - \mathbf{x}_{dmin}}{\mathbf{K}_s} \right\rfloor + 1, \tag{4.2}$$

kde $\overline{\mathbf{x}_{ind}} \in \mathbb{N}^{n_x}$ je vektor indexů diskrétních stavů reprezentace v Matlabu, $\mathbf{K}_s \in \mathbb{R}^{n_x}$ je vektor kvantizačních kroků pro jednotlivé dimenze stavu a $\mathbf{x}_{dmin} \in \mathbb{R}^{n_x}$ je vektor nejmenších hodnot jednotlivých stavů, pro které je vytvořena reprezentace v Matlabu. Jedničku je třeba přičíst z toho důvodu,

že Matlab indexuje od jedničky a ne od nuly jako standardní programovací jazyky. Dělení vektorů je zde myšleno po složkách. Konkrétní hodnoty diskrétních stavů se poté dají lehce přepočítat dle vztahu

$$\overline{\mathbf{x}} = \mathbf{x}_{dmin} + 0.5\mathbf{K}_s + \mathbf{K}_s(\overline{\mathbf{x}_{ind}} - 1). \tag{4.3}$$

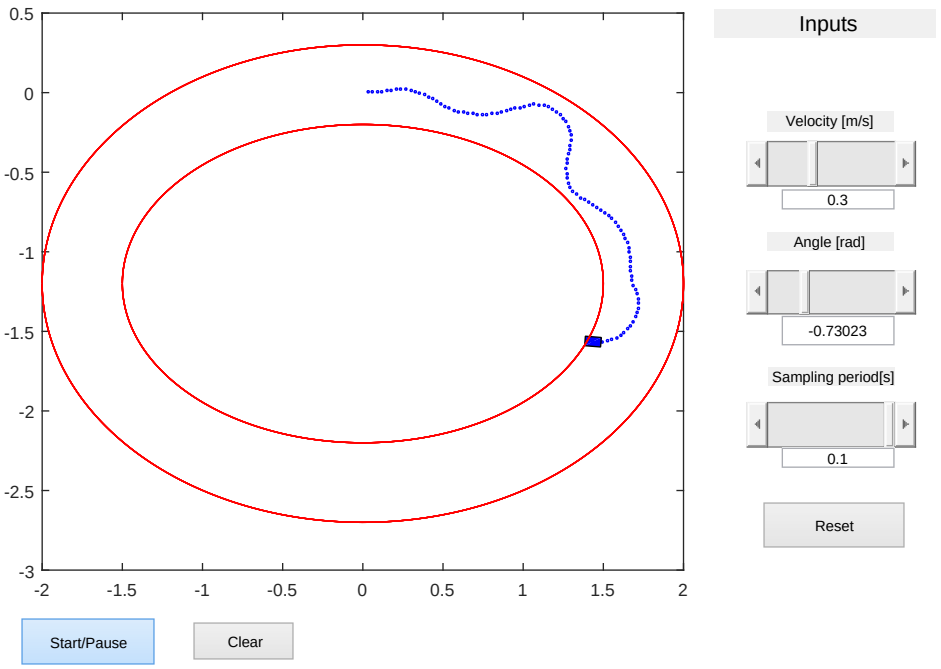
Rozdíly mezi kvantizací spojitého stavu pomocí agregační funkce (4.1) a agregační funkce realizované vztahy (4.3) a (4.2) jsou vidět na obrázku 4.1. Zatímco agregační funkce (4.1), nacházející se na obrázku 4.1 nahoře, mapuje hodnoty mimo kvantizační mřížku do krajních kvantizovaných stavů, agregační funkce (4.3), která bude v této práci dále používána, činí tyto hodnoty nepřípustné. Navíc je mapování těchto agregačních funkcí vůči sobě posunuté.



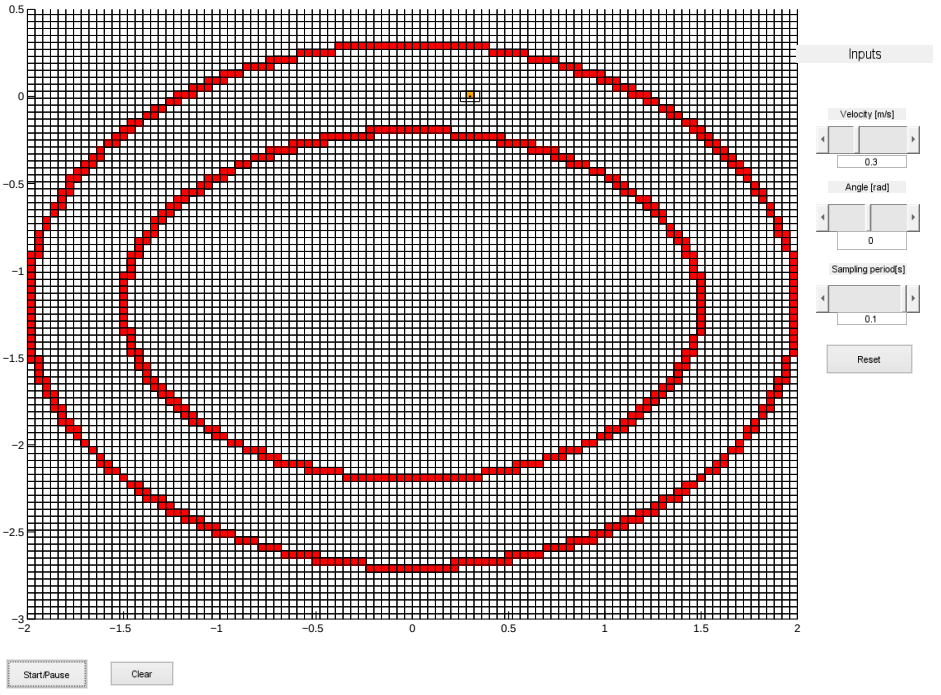
Obrázek 4.1: Porovnání dvou agregačních funkcí pro kvantizaci stavového prostoru

Na obrázku 4.2 lze vidět vytvořené grafické uživatelské rozhraní sloužící k ručnímu ovládání modelu RC auta. Dráha, po které se může model pohybovat, je reprezentována prostorem mezi červenými soustřednými elipsami. RC model auta je reprezentován modrým obdélníkem a modrá křivka reprezentuje trajektorii modelu v čase. Uživatel může nastavit rychlost a úhel natočení kol modelu, časový krok simulace a může také pozastavit, či obnovit simulaci do počátečního stavu.

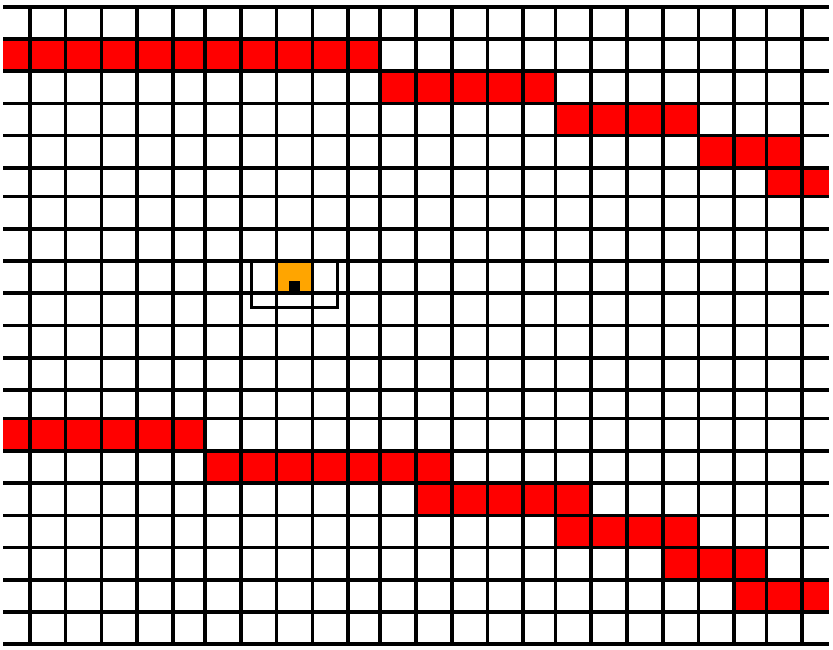
Toto grafické uživatelské rozhraní bylo dále rozšířeno tak, aby reflektovalo kvantizovaný prostoru. Vizualizace byla vylepšená o zobrazení průmětu spojitého stavu na diskrétní, jak je ilustrováno na obrázku 4.3 a v detailu na obrázku 4.4. Obě vizualizace se dají použít i pro simulaci a zobrazení optimální trajektorie nalezené pomocí dynamického programování.



Obrázek 4.2: Grafické uživatelské rozhraní



Obrázek 4.3: Ilustrace průmětu těžiště modelu na kvantizovaný bod mřížky stavového prostoru



Obrázek 4.4: Kvantizované a přiblížené grafické uživatelské rohraní

4.2 Matematický model prostředí

Začlenění matematického modelu prostředí do simulace lze provést vícero způsoby. Jednou z možností je změna kinematického pohybového modelu, kdy se například po kontaktu s překážkou mohou změnit pohybové rovnice. Matematický model prostředí lze také zakomponovat přímo do ztrátové funkce L^c, jejíž vhodnou volbou lze některé stavy upřednostnit před jinými, či je učinit nepřípustnými.

V této práci je model prostředí pro pohyb RC auta definován pomocí ztrátové funkce L^c. Tato ztrátová funkce hodnotí přípustné stavy libovolným reálným číslem, nepřípustné ∞ a cílovou pozici, nezávislé na úhlu natočení auta, hodnotí 0. Tím je trať jednoznačně určena, protože stavy s ohodnocením ∞ nepřipadají pro hledání optimálního řešení v úvahu.

4.3 Regulátor

V průběhu iterace váhové funkce se pro každý kvantizovaný stav ukládá optimální řízení do vektoru. Tato strategie řízení je poté použita pro simulaci a následnou vizualizaci pohybu auta. Při simulaci pohybu RC modelu použitím tohoto řízení ovšem nastává problém. RC model se ve skutečnosti nenachází

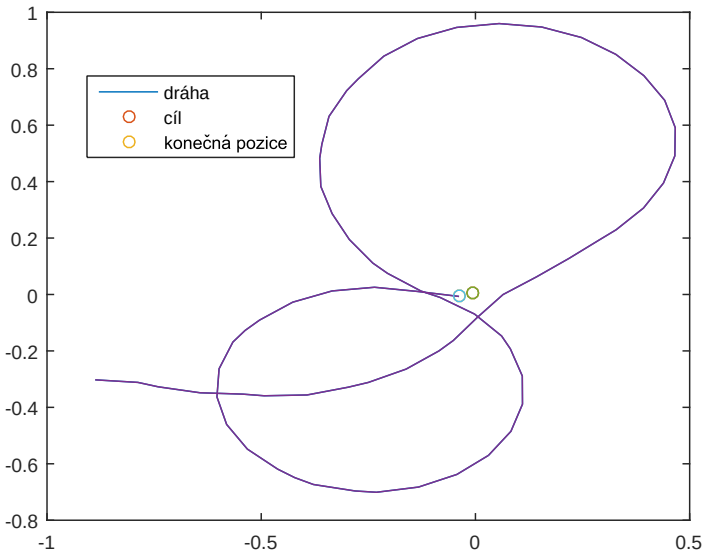
přesně v kvantizovaném stavu a je tedy možné, že po aplikaci řízení, které by ho mělo dostat do cíle, cíl mine a bude blíže nespecifikovaný čas jezdit kolem, viz obrázek 4.5 a 4.6.

Tomuto chování lze předejít tím, že při simulaci zvýšíme toleranci euklidovské vzdálenosti hodnotící dosažení cíle, oproti toleranci použité při návrhu řízení, viz obrázek 4.7. V tomto obrázku je použita maximální tolerance 0.08 oproti původním 0.05 v simulaci z obrázku 4.5, ale poté je třeba se smířit s tím, že RC auto nedojede přímo do cíle. V simulacích na obou obrázcích byla použita hrubá kvantizace (viz tabulka 4.1).

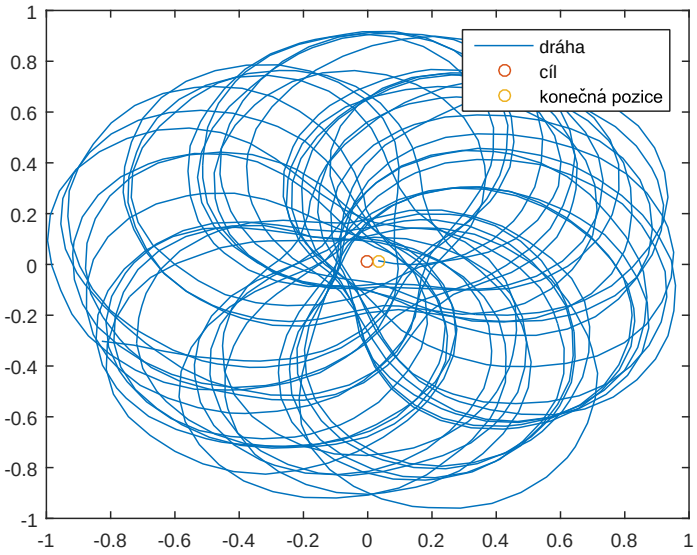
Druhou možností, jak problém zmírnit a přiblížit auto na konci dráhy k cílovému bodu, je zmenšit kvantizační krok. Nicméně například u jemné kvantizace (viz tabulka 4.1) vzniká 2 520 000 kvantizovaných stavů, což již představuje významné paměťové a výpočetní nároky. Na druhou stranu v případě užití této jemné kvantizace se dokážeme k cíli přiblížit až na 0.05m a pod lepším úhlem, viz obrázek 4.8.

Tabulka 4.1: Tabulka kvantizaci

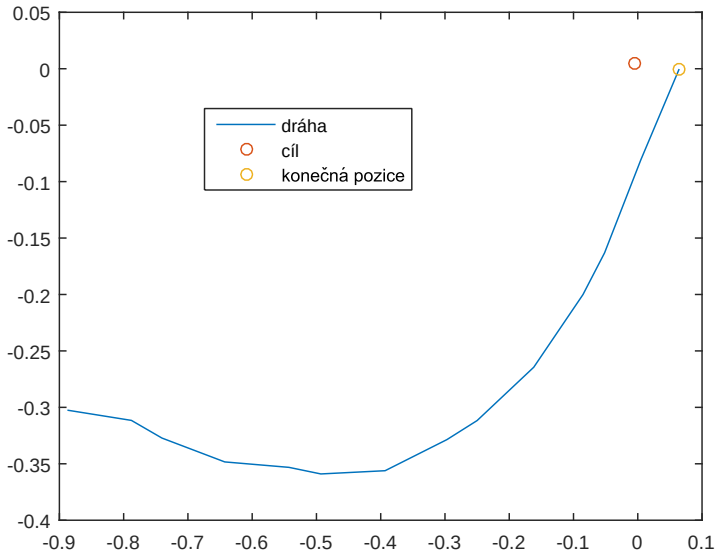
Kvantizace	x_1	x_2	x_3
hrubá	0.015	0.015	0.1
jemná	0.01	0.01	0.1



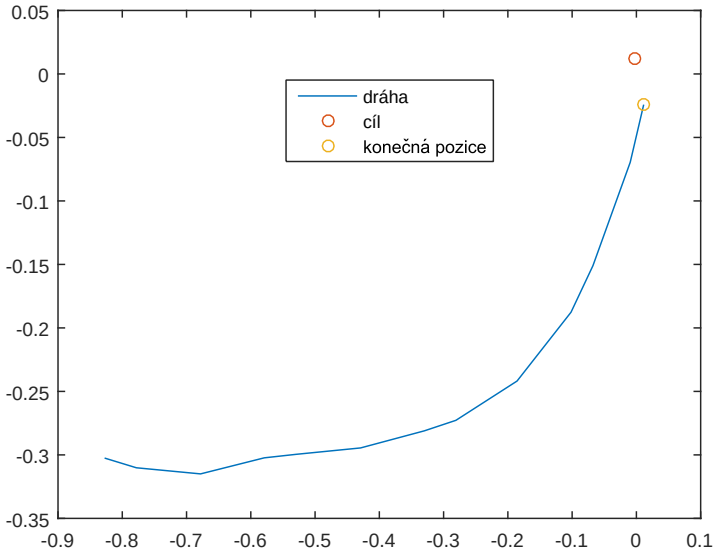
Obrázek 4.5: Trajektorie RC auta pro optimální řízení hrubou kvantizací a tolerancí dosažení cíle 0.05.



Obrázek 4.6: Trajektorie RC auta pro optimální řízení hrubou kvantizací a tolerancí dosažení cíle 0.04.



Obrázek 4.7: Trajektorie RC auta pro optimální řízení hrubou kvantizací a tolerancí dosažení cíle 0.08.



Obrázek 4.8: Trajektorie RC auta pro optimální řízení jemnou kvantizací a tolerancí dosažení cíle 0.05.

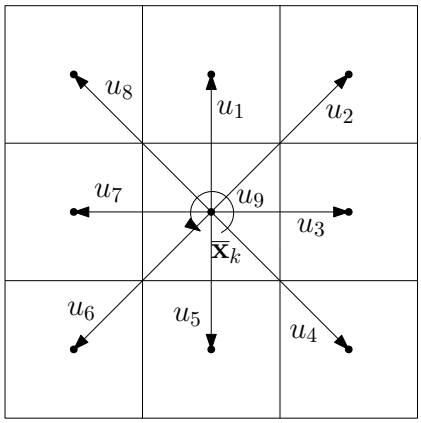
Kapitola 5

Simulační experimenty

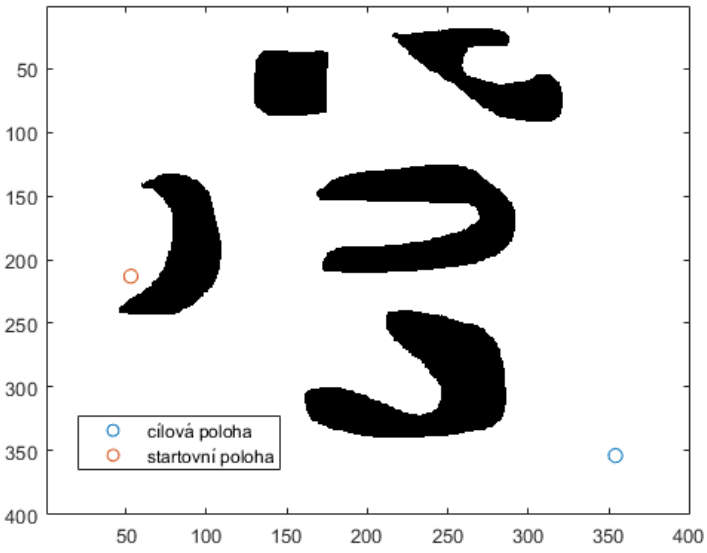
Tato část se věnuje několika simulačním experimentům, které slouží k ověření funkčnosti navrženého algoritmu a stanovení úprav, které bude zapotřebí provést, aby bylo možné nasadit algoritmus pro řízení reálného modelu RC auta. První simulační experiment navazuje a rozšiřuje příklad uvedený v kapitole 3.2.1. V druhém simulačním experimentu je již využit kinematický model pohybu modelu RC auta.

5.1 Nalezení nejkratší cesty

Oproti příkladu z podkapitoly 3.2.1 je zde uvažována šachovnice 400x400 políček, s překážkami znázorněnými černými plochami s L^c ohodnocením ∞, viz obrázek 5.2. Tato šachovnice byla vytvořena jako bitmapový obrázek a importována do Matlabu. Možná řízení jsou rozšířením řízení z předchozího příkladu na iteraci váhové funkce a jsou zobrazena na obrázku 5.1.



Obrázek 5.1: Růžice řízení

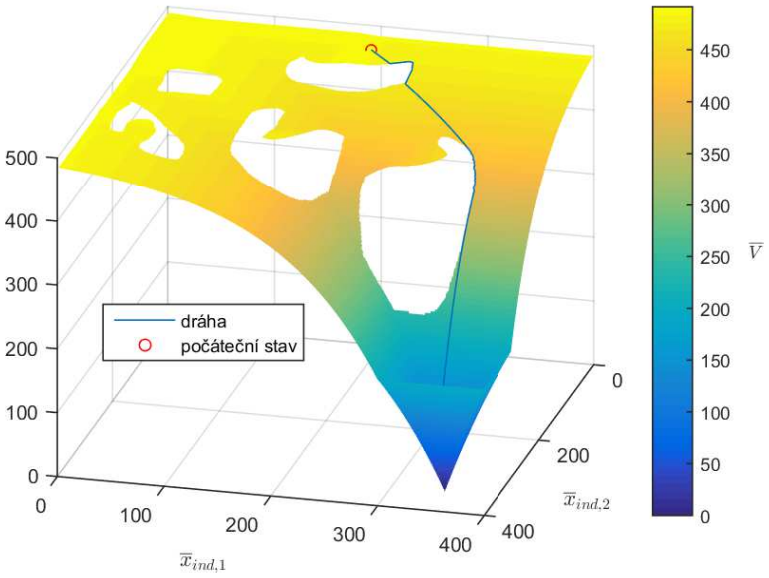


Obrázek 5.2: Šachovnice 400x400

Parametry simulace

Počáteční stav: $[-0.83, -0.31, 0]$m
Koncový stav: $[-0.09, 0.09]$m
Krok kvantizační mřížky: 0.05m
Krok simulace: 0.1s
Přípustné rychlosti: $[0.5, 1]$m/s
Přípustné natočení koleček: $-0.3, -0.28, ..., 0.3$rad
maximální vzdálenost od cíle = 0.07cm
l = 0.55m
$\eta = 0.99$
$\delta = 0.0001$

Bellmanova funkce nalezená pomocí algoritmu iterace váhové funkce je graficky zobrazena na obrázku 5.3. Hodnota funkce roste ve všech směrech od cílového bodu a v místě překážek má hodnotu ∞. Na obrázku je dále vykreslen startovní bod společně s optimální trajektorií. Tato trajektorie je ve skutečnosti složena pouze ze dvou dimenzí, ale pro větší názornost se její třetí souřadnice sestává z hodnot Bellmanovy funkce, kterými právě prochází.



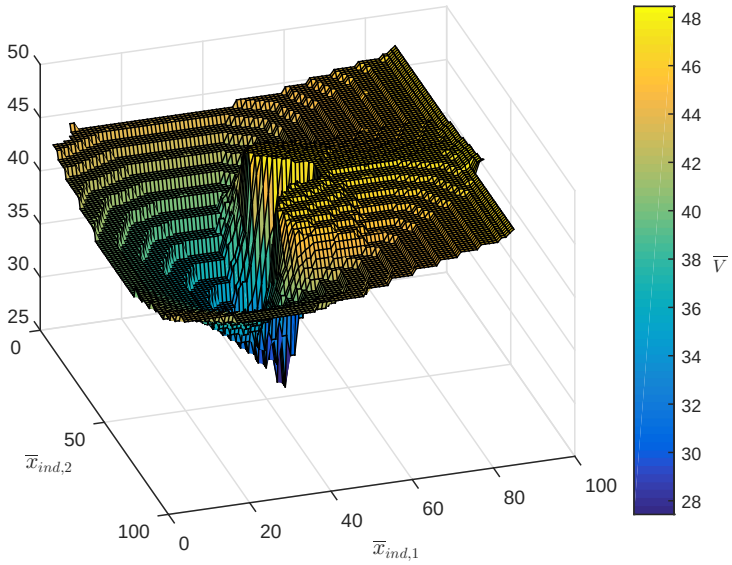
Obrázek 5.3: Bellmanova funkce se zvýrazněným optimálním pohybem

5.2 Řízení RC modelu auta

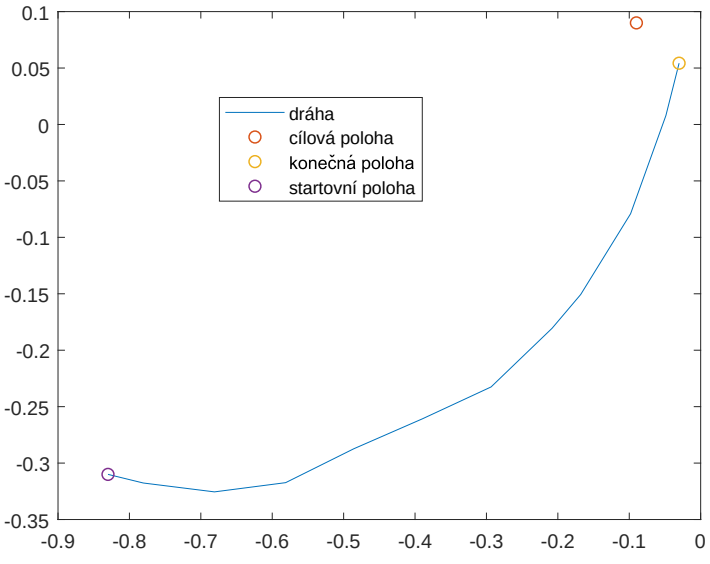
Druhý simulační experiment již využívá kinematický model pohybu modelu RC auta. Pro snížení výpočetních nároků iterace váhové funkce, a tím pádem ušetření času, byla zvolena ztrátová funkce L^c taková, že hodnotí všechny stavy v předem dané čtvercové oblasti libovolným reálným číslem, okolí opět ∞ a cílový stav 0, to značně snižuje čas potřebný pro iteraci váhové funkce.

Parametry simulace

Počáteční stav: $[-0.83, -0.31, 0]$m
Koncový stav: $[-0.09, 0.09]$m
Krok kvantizační mřížky: 0.02m pro x_1, x_2 a 0.05rad pro x_3
Krok simulace: 0.1s
Přípustné rychlosti: $[0.5, 1]$m/s
Přípustné natočení koleček: $-0.3, -0.28, ..., 0.3$rad
maximální vzdálenost od cíle = 0.07cm
l = 0.55m
$\eta = 0.9$
$\delta = 0.01$



Obrázek 5.4: Bellmanova funkce pro $x_3 = 0$



Obrázek 5.5: Trajektorie RC auta

Obrázek 5.4 zobrazuje Bellmanovu funkci pro úhel natočení RC auta $x_3 = 0$. Lze na něm vidět jak ohodnocení Bellmanovy funkce směrem k cíli klesá pouze z jedné strany, díky omezením zaneseným do pohybu RC auta přidáním jeho dynamiky. Skok v hodnotách Bellmanovy funkce vzniká ve chvíli, kdy je RC auto již moc blízko cíli a nedokáže do něj zatočit přímo, ale musí se vrátit. Na obrázku 5.5 je znázorněna dráha modelu RC auta řízeného pomocí navrženého regulátoru. Je zde také vidět, že RC auto nedojede přímo do cíle, díky problému s kvantizací, viz kapitola 4.3.

Vzhledem k problému s kvantizací, popsanému v kapitole 4.3, by pro reálné nasazení bylo pravděpodobně třeba zmenšit kvantizační krok. Toto zmenšení však značně zvýší výpočetní nároky. Další možností by bylo hledat spojitou Bellmanovu funkci ve spojitém prostoru.

Kapitola 6

Závěr

Cílem této práce bylo seznámit se s dynamickým programováním a tuto znalost následně využít k vytvoření algoritmu pro nalezení optimálního řízení RC modelu auta, s ohledem k budoucímu nasazení na reálný model. Pro naplnění tohoto cíle bylo třeba se okrajově zabývat několika tématy.

Nejprve byly zjištěny parametry reálného modelu RC auta, aby mohly být simulace a výpočty prováděny s parametry odpovídajícími skutečnosti. Dalším krokem bylo nalezení vhodného matematického modelu popisujícího pohyb RC auta, vzhledem k tomu, že tato práce pouze prozkoumává možnosti řízení reálného RC modelu, postačuje i přibližný matematický model. Proto byl zvolen zjednodušený kinematický model odpovídající modelu jízdního kola. Dále bylo porovnáno několik základních metod diskretizace a diskutována jejich vhodnost použití v simulaci pro tuto práci a použití při hledání optimálního řízení pro reálný model RC auta. Pro tuto práci byla, vzhledem k výše zmíněným důvodům zvolena ta implementačně nejméně strojově náročná.

Následovalo nastudování teorie dynamického programování pro konečný a nekonečný časový horizont. Pro nekonečný časový horizont byl poté vytvořen jednoduchý příklad. Dalším krokem již bylo navržení samotného algoritmu hledajícího optimální řízení. Stavový prostor bylo nejprve nutno kvantizovat, aby mohla být jednodušeji reprezentována Bellmanova funkce. Hlavním kritériem při implementaci kvantizace byla opět rychlost. V poslední řadě bylo třeba vybrat vhodný matematický model prostředí pohybu RC modelu. Následovala diskuse o problémech optimálního řízení pro kvantizovaný prostor. Nakonec bylo provedeno několik simulačních experimentů, které ověřily funkčnost navrženého algoritmu a umožnily zhodnotit úpravy potřebné pro jeho použití při hledání optimální strategie řízení pro reálný model RC auta.

Navazující práce by se mohla zajímat převážně aplikací již hotového algoritmu na reálný model RC auta, k tomu by bylo třeba zpřesnit výpočet op-

timálního řízení, pravděpodobně použitím velmi jemné kvantizace. Možností by také bylo nekvantizovat prostor a pokusit se reprezentovat Bellmanovu spojitou funkci. Pro aplikaci na reálný model by také bylo třeba nalézt vztah mezi řízením natočení kol v radiánech a rychlosti v metrech za sekundu a řízením akutátorů přímo v reálnem modelu RC auta.

Seznam obrázků

Literatura

[1] Erik Cheever. Approximation of differential equations by numerical integration. http://lpsa.swarthmore.edu/NumInt/NumIntIntro.html, 2005. Accessed: 2018-02-23.

[2] Dr. M. Gerdts. The single track model. https://www.scribd.com/document/59767869/The-Single-Track-Model, 2003. Accessed: 2018-03-18.

[3] William Press, Saul Teukolsky, William Vetterling, and Brian Flannery. *Numerical Recipes. The Art of Scientific Computing*. Cambridge university press, United Kingdom, 3st edition, 2007. ISBN 9780521880688.

[4] Rajesh Rajamani. *Vehicle Dynamics and Control*. Springer, New York, NY, USA, 2nd edition, 2012. ISBN 978-1-4614-1433-9.

[5] Dieter Schramm, Manfred Hiller, and Roberto Bardini. *Vehicle Dynamics, Modeling and Simulation*. Springer, Heidelberg, New York, Dordrecht, London, 1st edition, 2014. ISBN 978-3-540-36045-2.

[6] Lawrence Shampine and Mark Reichelt. The matlab ode suite. https://www.mathworks.com/help/pdf_doc/otherdocs/ode_suite.pdf. Accessed: 2018-02-28.

[7] Draguna Vrabie, Kyriakos Vamavoudakis, and Frank Lewis. *Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles*. The Institution of Engineering and Technology, London, United Kingdom, 1st edition, 2012. ISBN 978-1-84919-489-1.